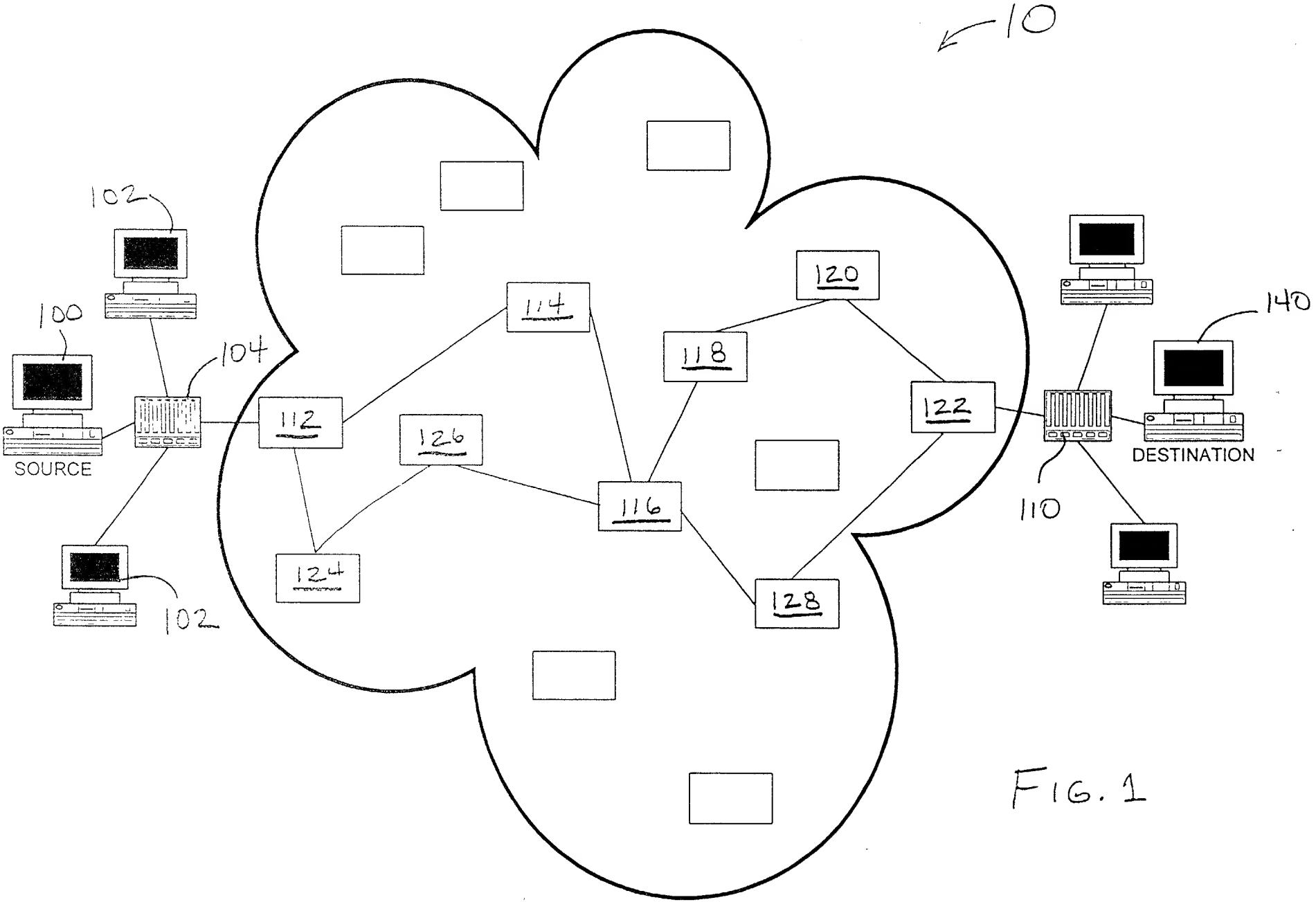

10
102
100
104
112
114
120
118
122
140
126
116
110
124
128
102
SOURCE
DESTINATION

FIG. 1

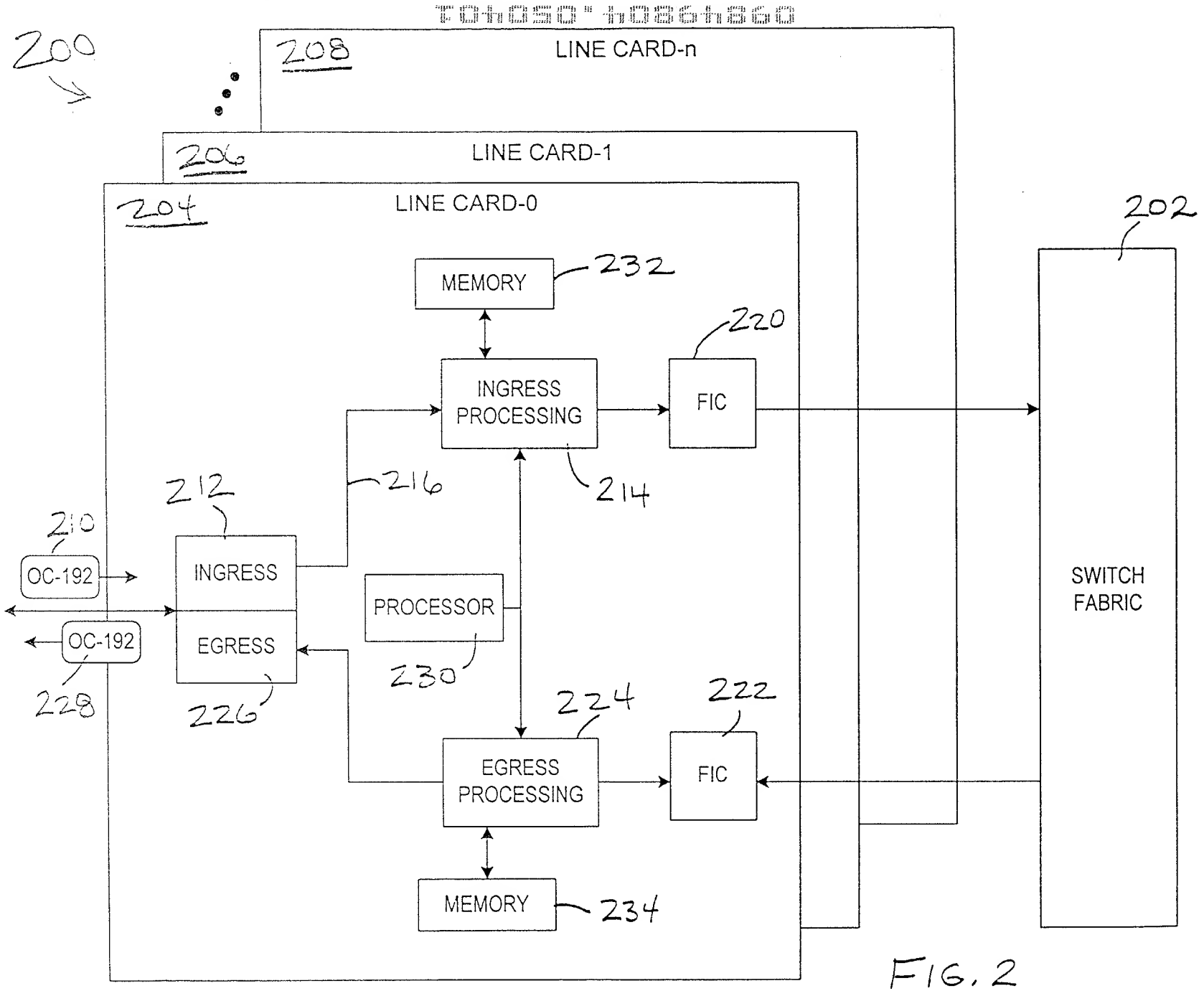
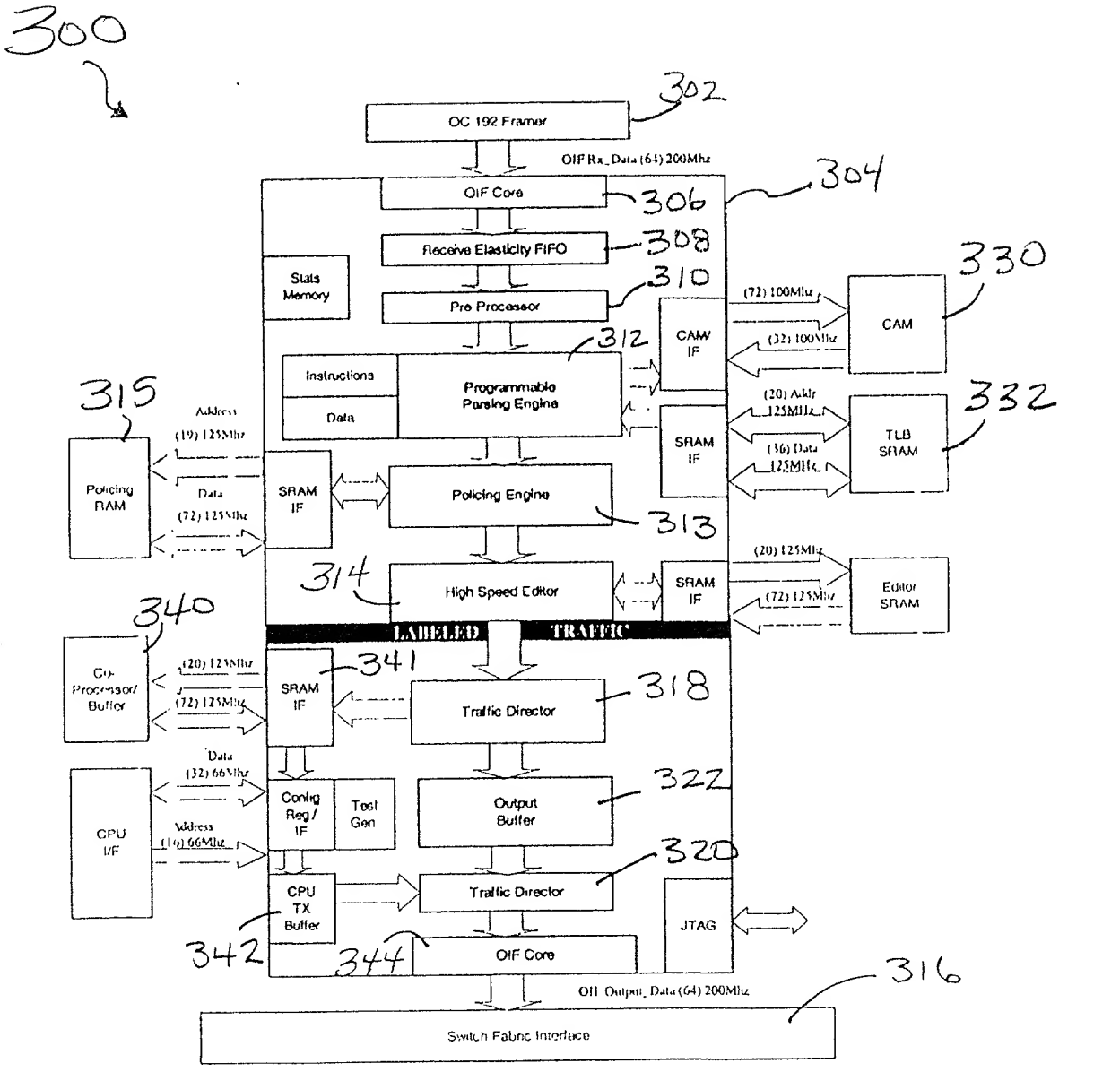

200
208
LINE CARD-n
206
LINE CARD-1
204
LINE CARD-0
202
MEMORY
232
220
INGRESS PROCESSING
FIC
216
214
212
210
OC-192
INGRESS
PROCESSOR
SWITCH FABRIC
OC-192
EGRESS
230
228
226
224
222
EGRESS PROCESSING
FIC
MEMORY
234

FIG. 2

FIG. 3


300
302
OC 192 Framer
OIF Rx_Data (64) 200Mhz
304
OIF Core
306
Receive Elasticity FIFO
308
Stats Memory
310
Pre Processor
312
Instructions
Data
Programmable Parsing Engine
330
CAM
CAM IF
(72) 100Mhz
(32) 100Mhz
332
TLB SRAM
SRAM IF
(20) Addr 125MHz
(36) Data 125MHz
315
Policing RAM
Address (19) 125Mhz
Data (72) 125Mhz
SRAM IF
Policing Engine
313
314
High Speed Editor
SRAM IF
(20) 125Mhz
(72) 125Mhz
Editor SRAM
LABELED
TRAFFIC
340
Co-Processor/ Buffer
(20) 125Mhz
(72) 125Mhz
SRAM IF
341
Traffic Director
318
Data (32) 66Mhz
Address (19) 66Mhz
CPU I/F
Config Reg / IF
Test Gen
Output Buffer
322
CPU TX Buffer
342
Traffic Director
320
JTAG
344
OIF Core
OIF Output_Data (64) 200Mhz
316
Switch Fabric Interface

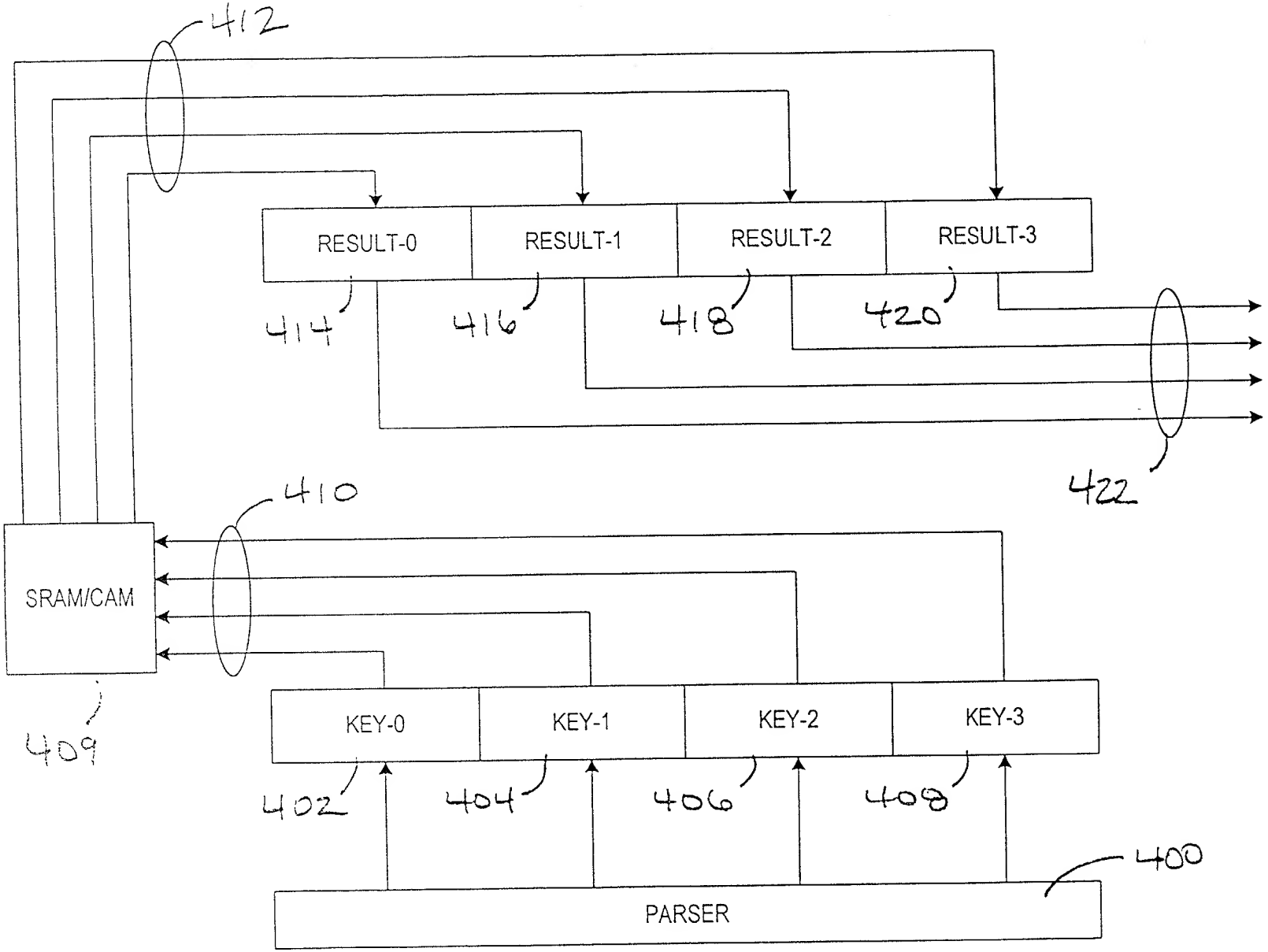

412
RESULT-0
RESULT-1
RESULT-2
RESULT-3
414
416
418
420
422
410
SRAM/CAM
409
KEY-0
KEY-1
KEY-2
KEY-3
402
404
406
408
400
PARSER

FIG. 4

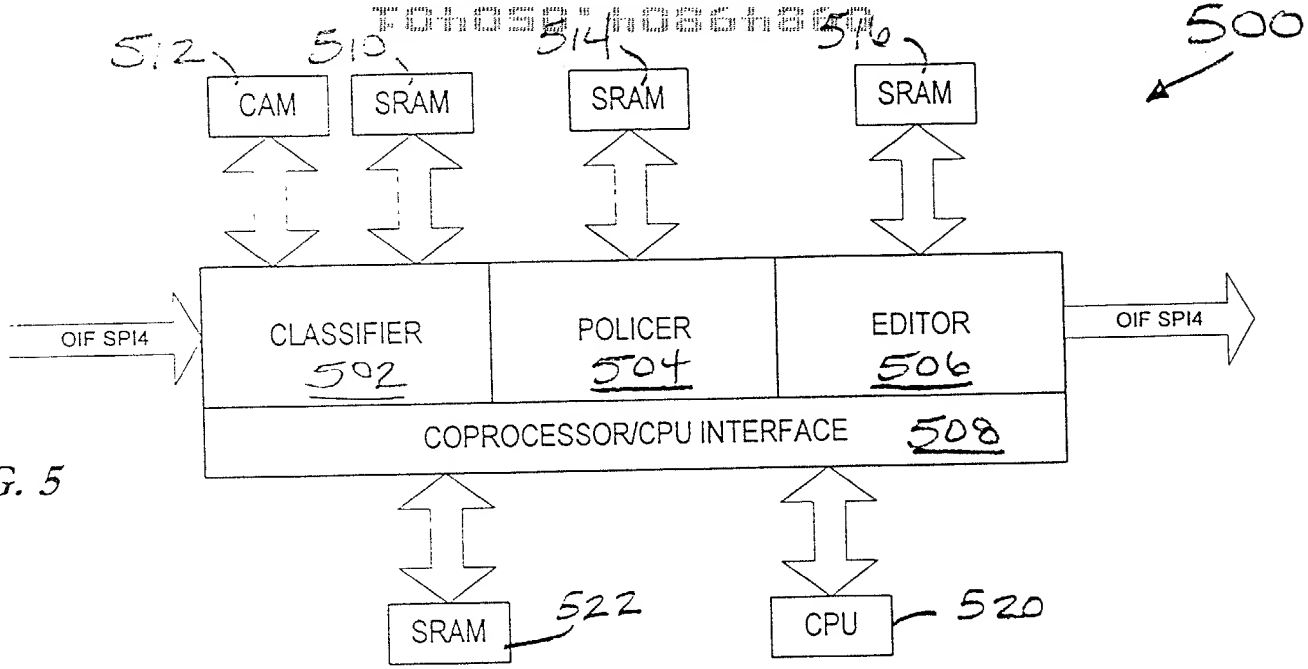

500
512
CAM
510
SRAM
514
SRAM
516
SRAM
OIF SPI4
CLASSIFIER
502
POLICER
504
EDITOR
506
OIF SPI4
COPROCESSOR/CPU INTERFACE
508
SRAM
522
CPU
520

FIG. 5

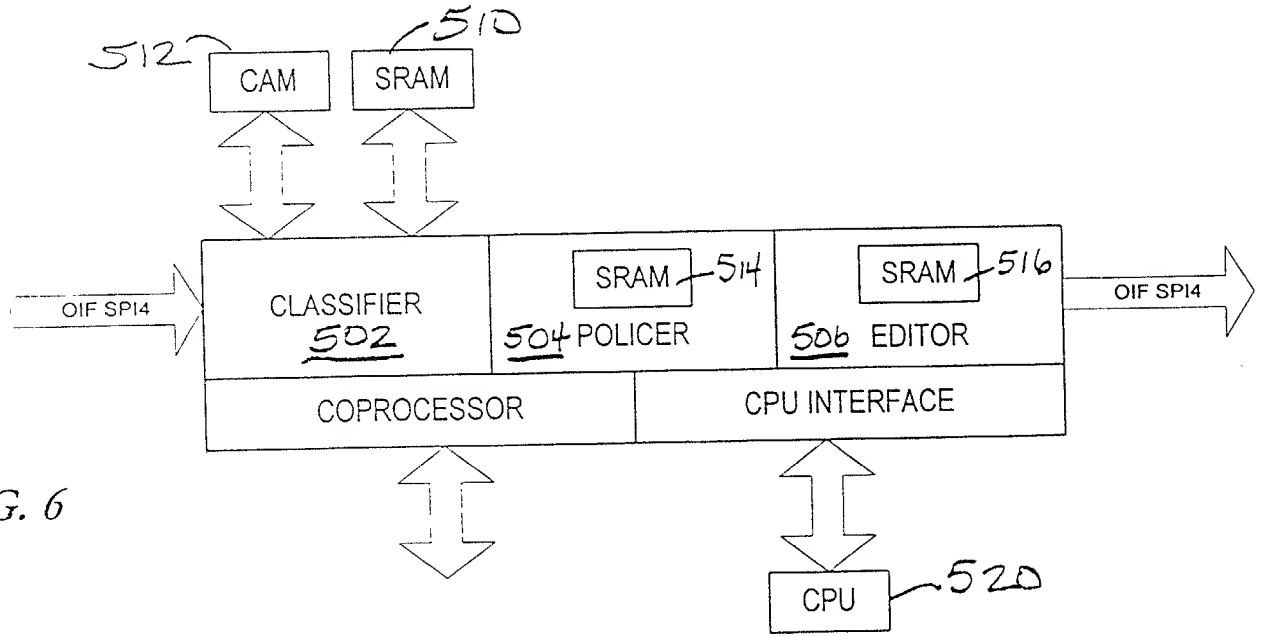

512
CAM
510
SRAM
OIF SPI4
CLASSIFIER
502
SRAM
514
504 POLICER
SRAM
516
506 EDITOR
OIF SPI4
COPROCESSOR
CPU INTERFACE
CPU
520

FIG. 6

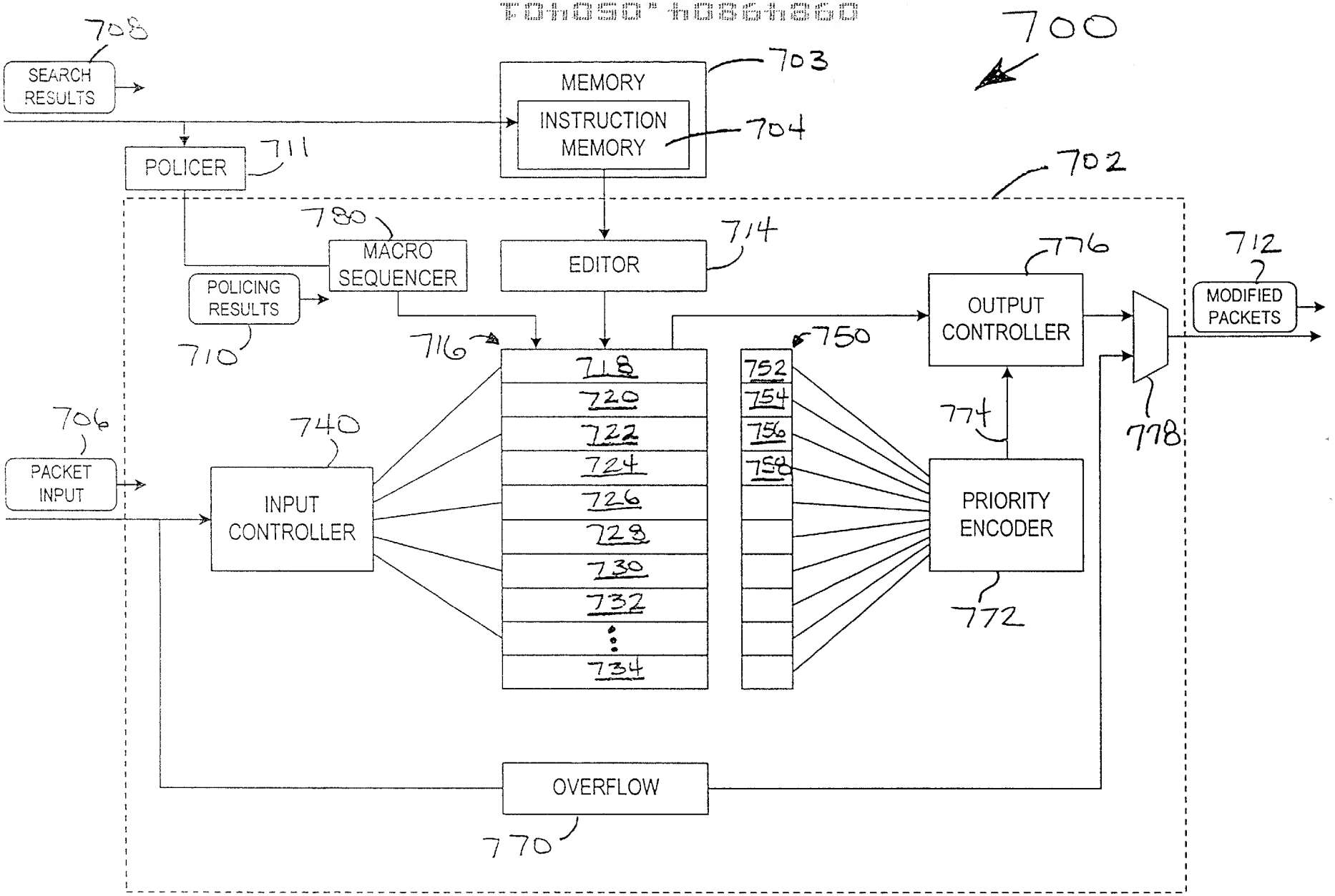


FIG. 7

800

Editor Instruction	Description
No-Op	No operation is performed
Write1	General purpose single word write
Write2	General purpose double word write
Delete1	General purpose single word delete
Delete2	General purpose double word delete
Read-Modify-Write with Mask	Single word modify; mask provided with data
Read-Modify-Write with Default Mask	Mask provided from one of a plurality of default masks
Swap	Swap the current top MPLS label
Swap/Push1	Swap the current top MPLS label and push one other to the top
Push1	Push one MPLS label to the top of the MPLS stack
Swap/Push2	Swap the current top MPLS label and push two others to the top
Push2	Push two MPLS labels to the top of the MPLS stack
Pop1/Swap	Pop the current top MPLS label and swap the next
Pop1	Pop the current top MPLS label
Pop2	Pop the two current top MPLS labels
PopAll	Pop all MPLS labelsfrom stack

FIG. 8

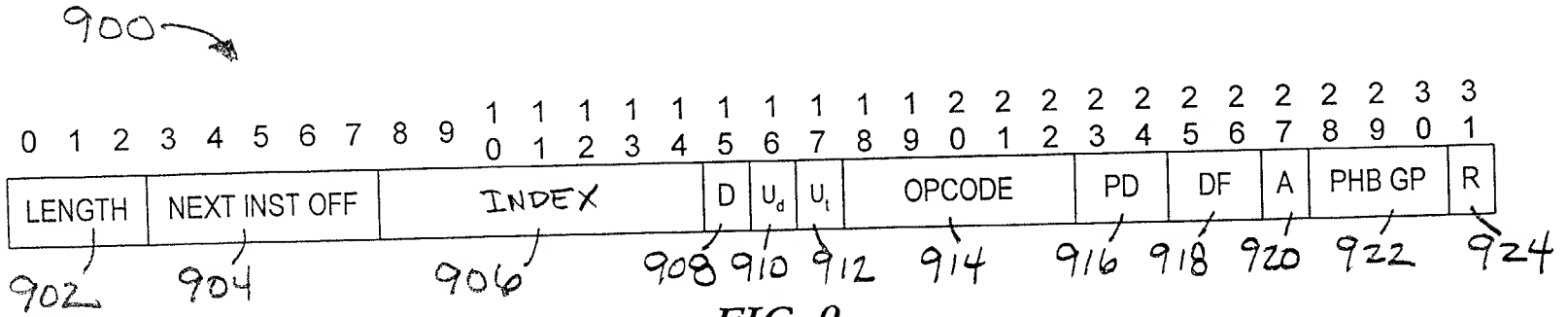


FIG. 9

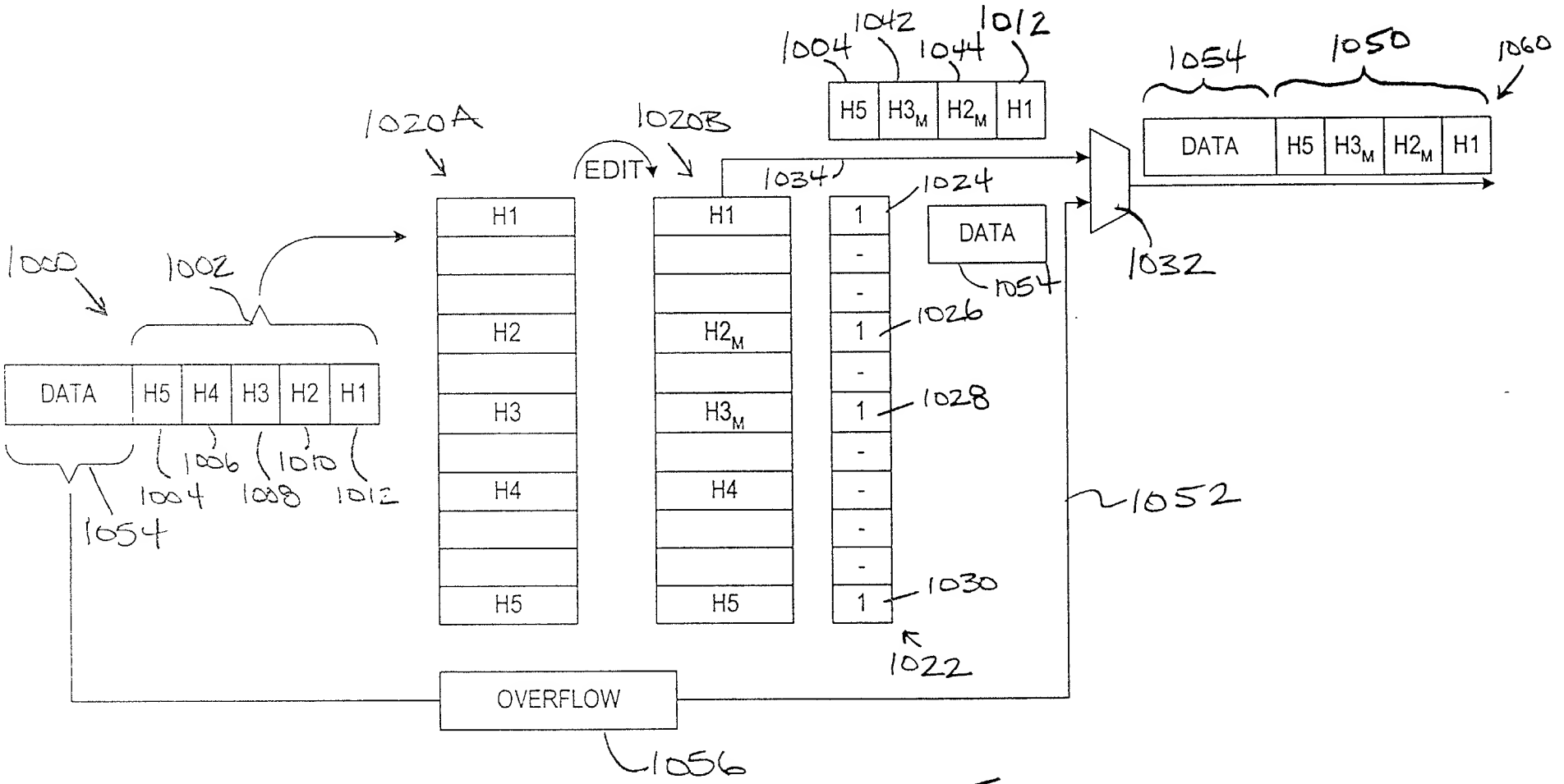
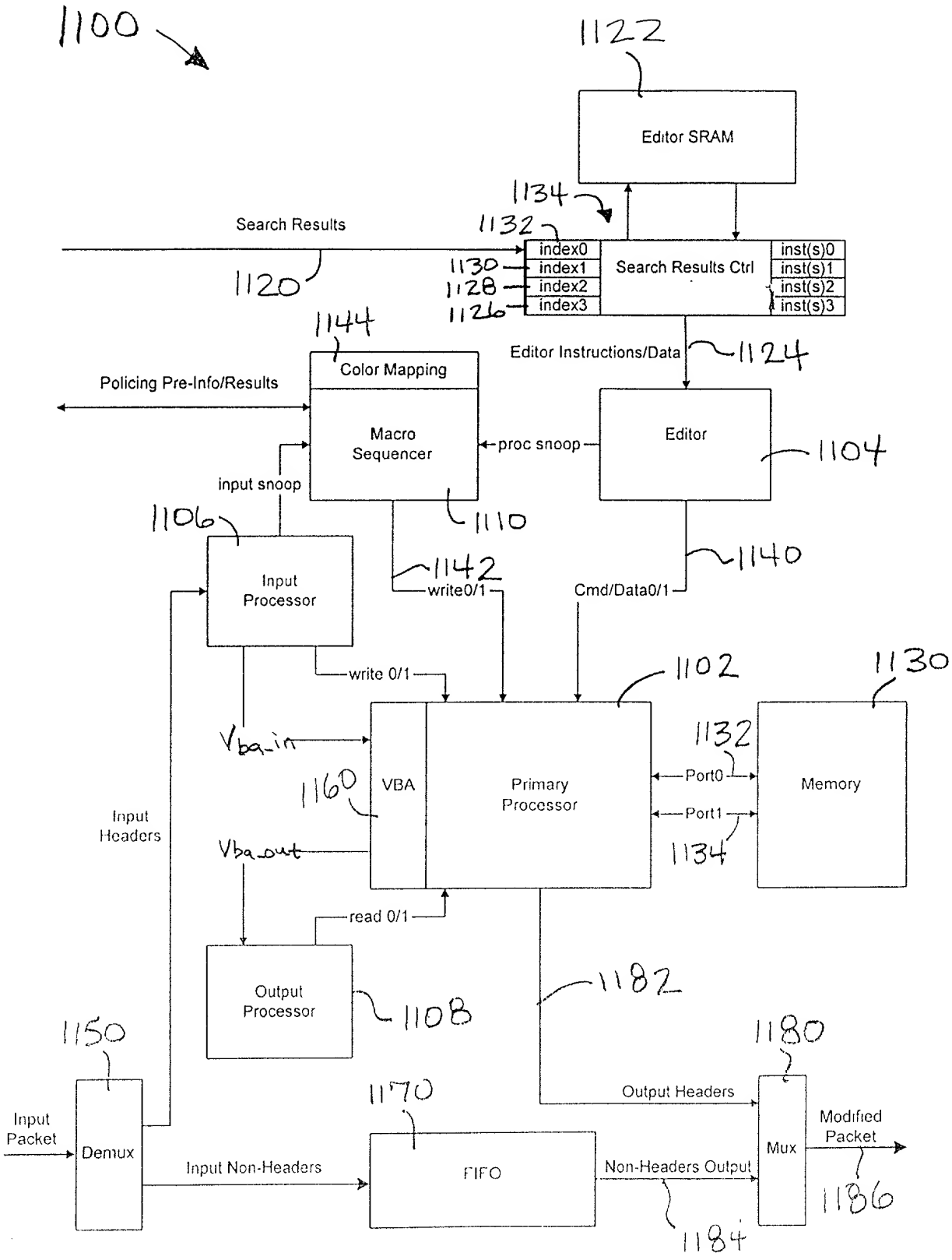
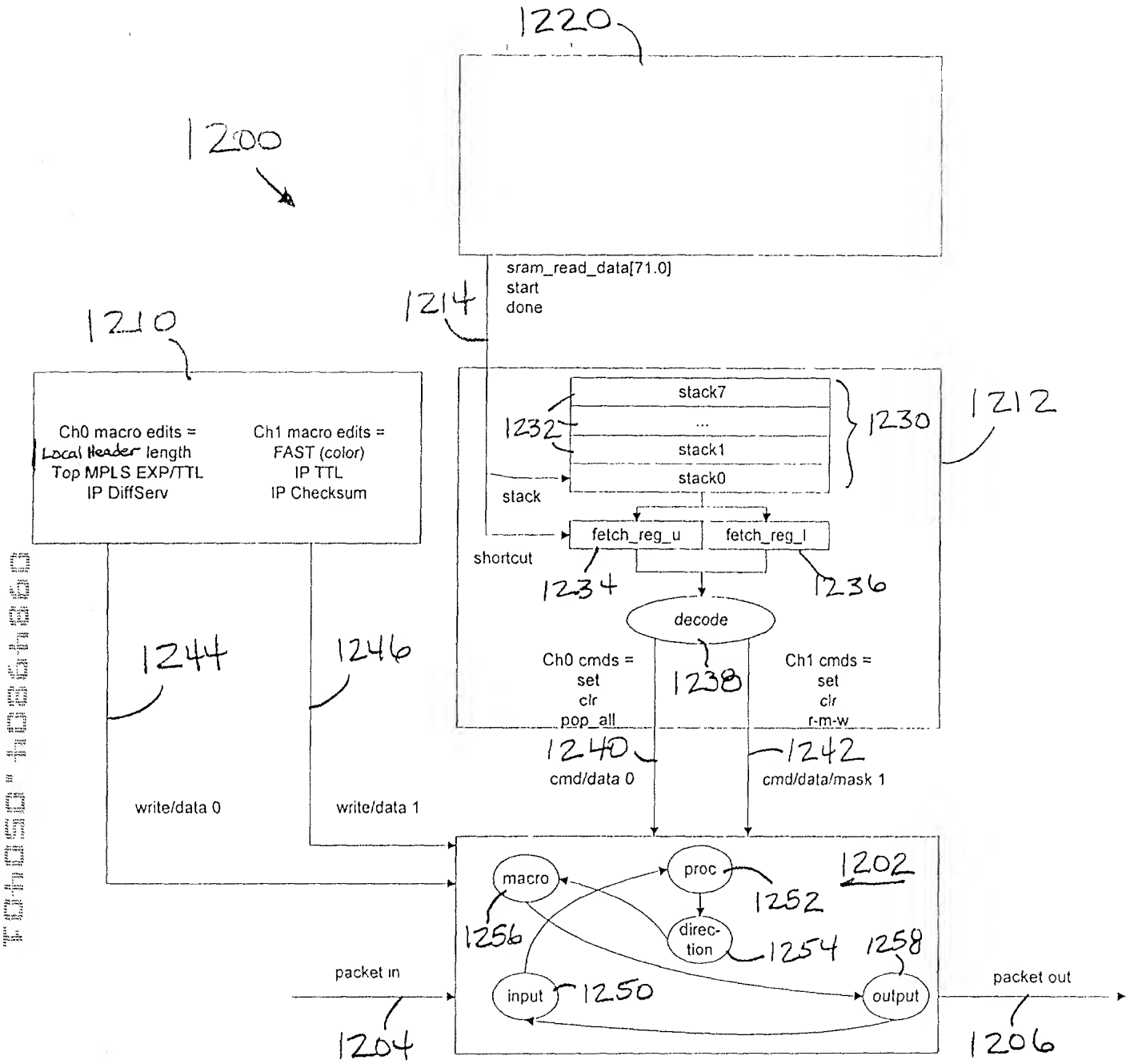

1000
1002
DATA
H5
H4
H3
H2
H1
1004
1006
1008
1010
1012
1054
1020A
1020B
EDIT
H1
H2
H3
H4
H5
H2M
H3M
1034
1024
1026
1028
1030
1022
1042
1044
1012
1004
H5
H3M
H2M
H1
DATA
1054
1032
1050
1060
1052
OVERFLOW
1056

FIG. 10


1100
1122
Editor SRAM
1134
1132
Search Results
1120
index0
index1
index2
index3
1130
1128
1126
Search Results Ctrl
inst(s)0
inst(s)1
inst(s)2
inst(s)3
1144
Color Mapping
Editor Instructions/Data
1124
Policing Pre-Info/Results
Macro Sequencer
proc snoop
Editor
1104
input snoop
1106
1110
1140
Input Processor
1142
write0/1
Cmd/Data0/1
write 0/1
1102
1130
Vba_in
1132
Port0
Memory
1160
VBA
Primary Processor
Port1
1134
Input Headers
Vba_out
read 0/1
1182
Output Processor
1108
1150
1180
Output Headers
1170
Input Packet
Demux
Input Non-Headers
FIFO
Non-Headers Output
Mux
Modified Packet
1186
1184


1220
1200
sram_read_data[71.0]
start
done
1214
1210
1212
Ch0 macro edits =
Local Header length
Top MPLS EXP/TTL
IP DiffServ
Ch1 macro edits =
FAST (color)
IP TTL
IP Checksum
stack7
...
stack1
stack0
1232
1230
stack
fetch_reg_u
fetch_reg_l
shortcut
1234
1236
decode
1238
Ch0 cmds =
set
clr
pop_all
Ch1 cmds =
set
clr
r-m-w
1240
1242
cmd/data 0
cmd/data/mask 1
1244
1246
write/data 0
write/data 1
1202
macro
proc
1252
direc-
tion
1254
1256
1258
packet in
input
1250
output
packet out
1204
1206

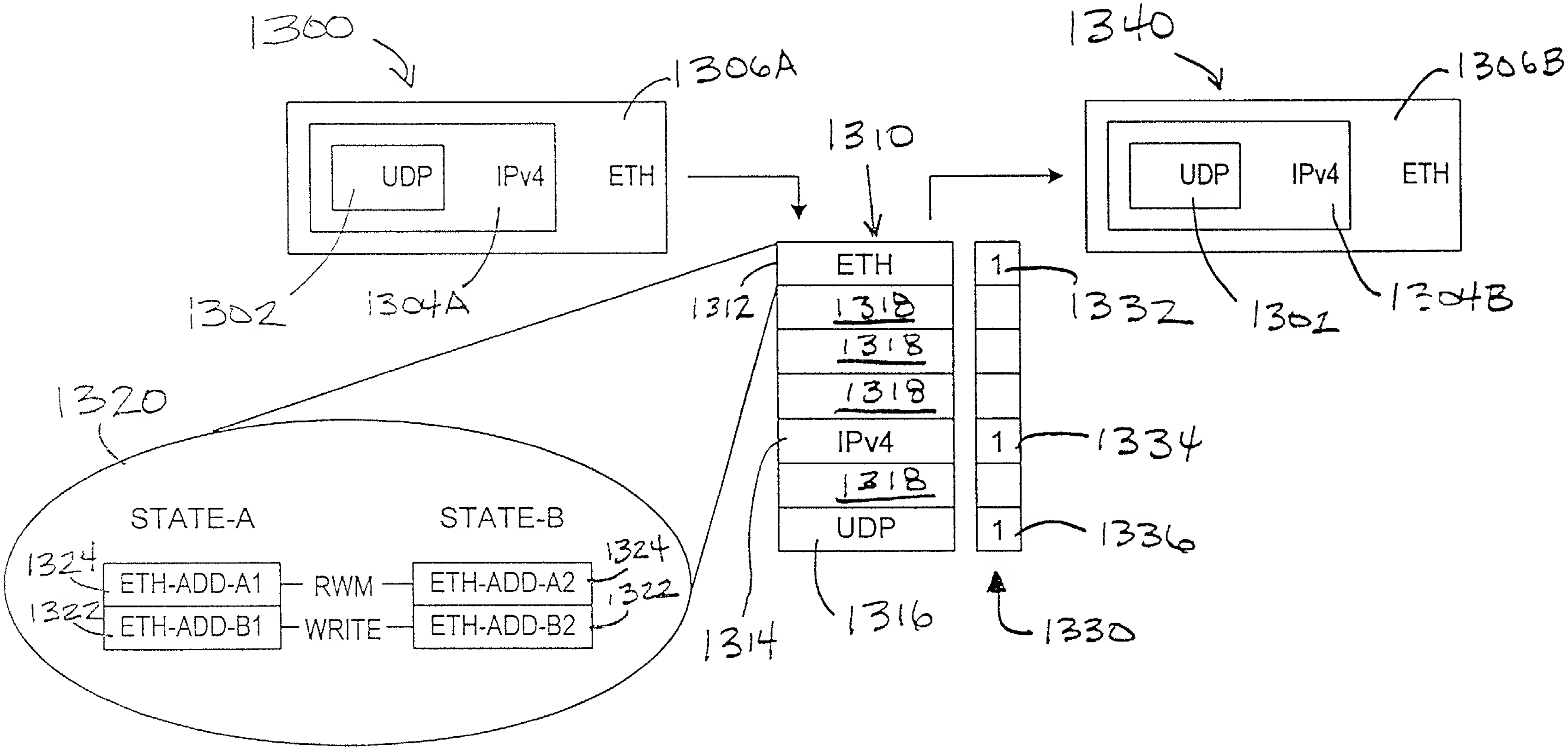

1300
1306A
UDP
IPv4
ETH
1302
1304A
1310
ETH
1318
1318
1318
IPv4
1318
UDP
1312
1314
1316
1
1
1
1332
1334
1336
1330
1340
1306B
UDP
IPv4
ETH
1302
1304B
1320
STATE-A
STATE-B
1324
1322
ETH-ADD-A1
ETH-ADD-B1
RWM
WRITE
ETH-ADD-A2
ETH-ADD-B2
1324
1322

FIG. 13

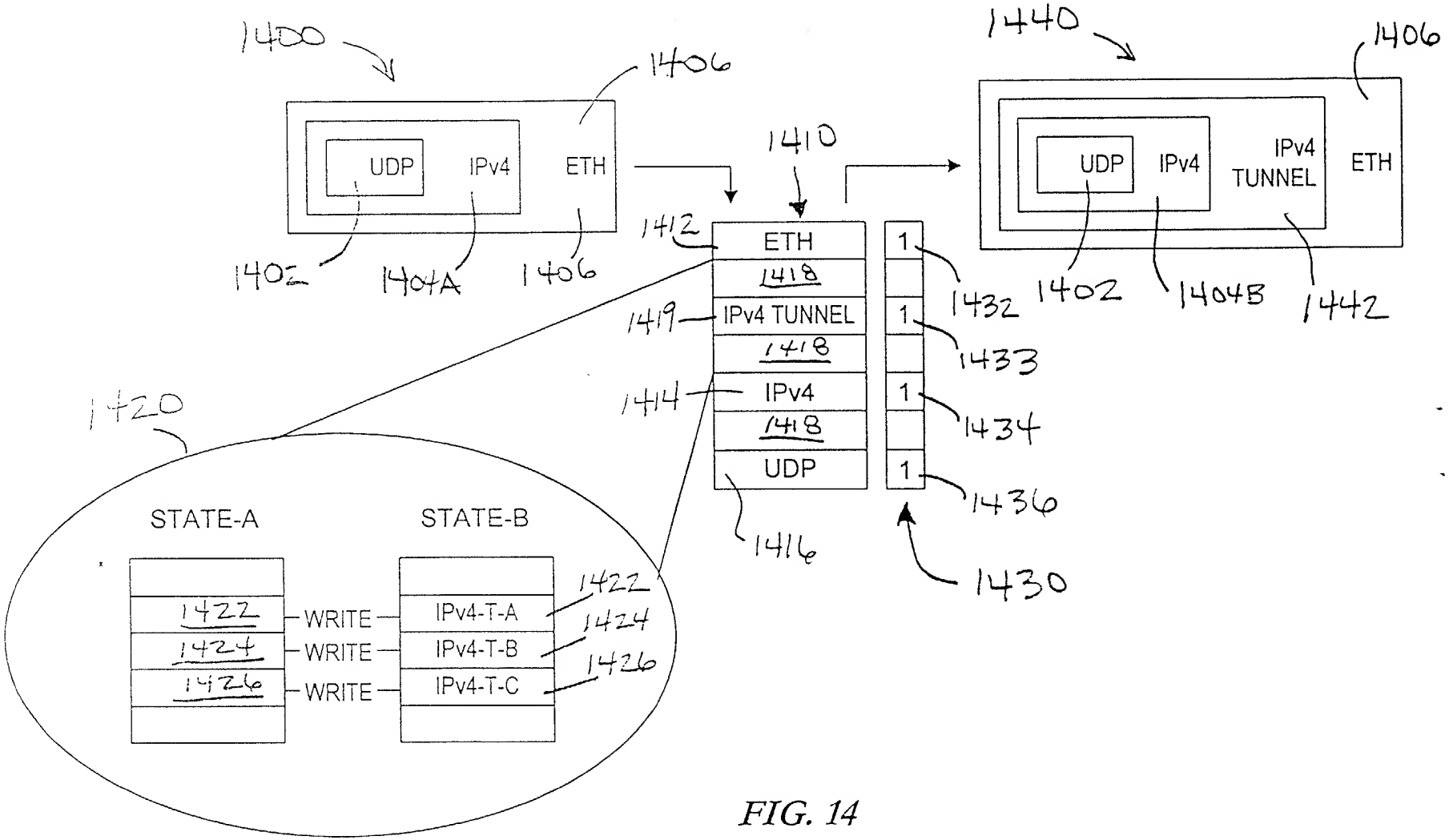

1400
1406
UDP
IPv4
ETH
1402
1404A
1406
1410
1412
ETH
1418
1419
IPv4 TUNNEL
1418
1414
IPv4
1418
UDP
1416
1
1432
1
1433
1
1434
1
1436
1430
1440
1406
UDP
IPv4
IPv4 TUNNEL
ETH
1402
1404B
1442
1420
STATE-A
STATE-B
1422
WRITE
IPv4-T-A
1424
1424
WRITE
IPv4-T-B
1426
1426
WRITE
IPv4-T-C

FIG. 14

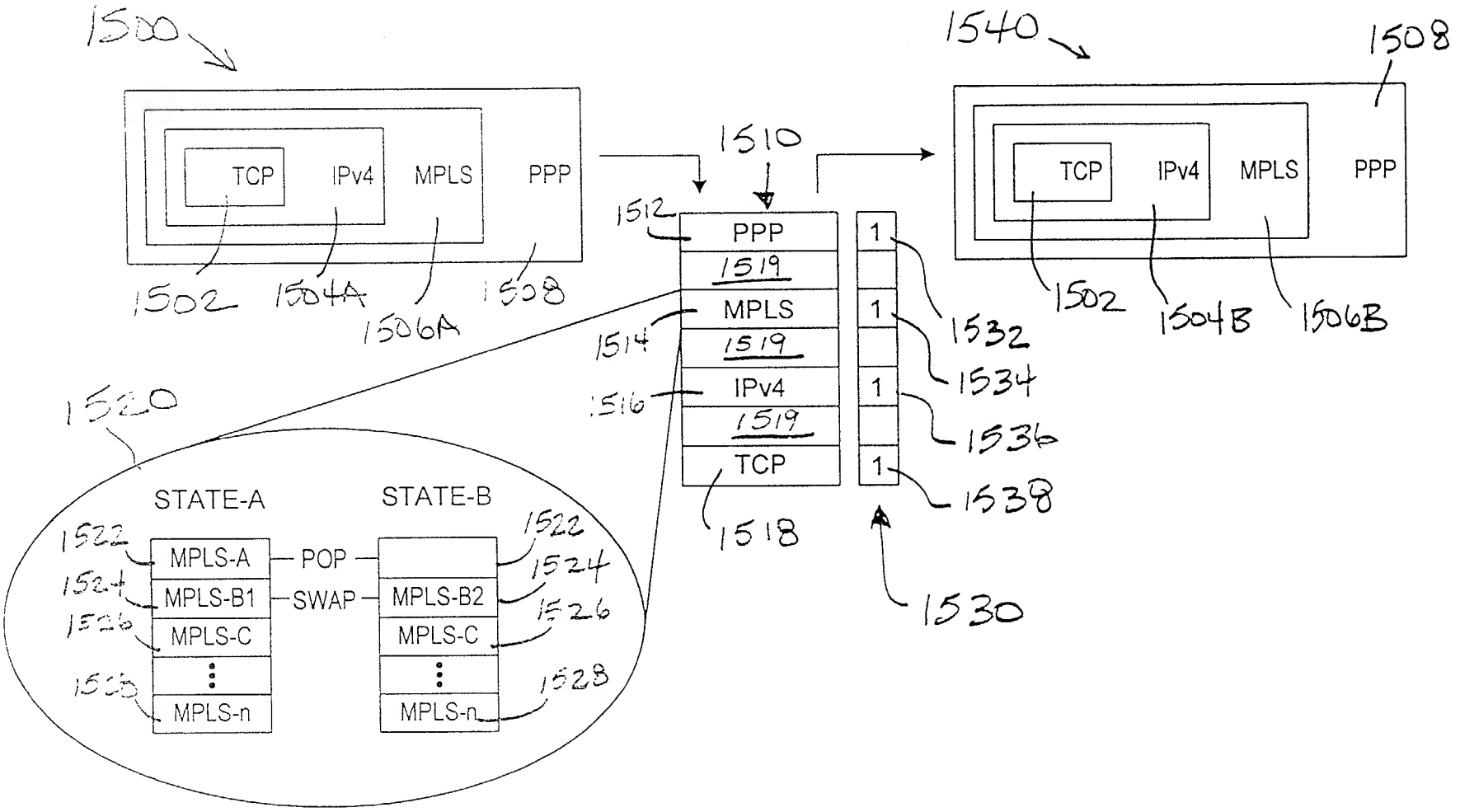

1500
TCP
IPv4
MPLS
PPP
1502
1504A
1506A
1508
1510
1512
PPP
1519
MPLS
1519
IPv4
1519
TCP
1514
1516
1518
1520
STATE-A
STATE-B
MPLS-A
POP
MPLS-B1
SWAP
MPLS-C
MPLS-n
MPLS-B2
MPLS-C
MPLS-n
1522
1524
1526
1528
1530
1532
1534
1536
1538
1540
1502
1504B
1506B
1508

FIG. 15

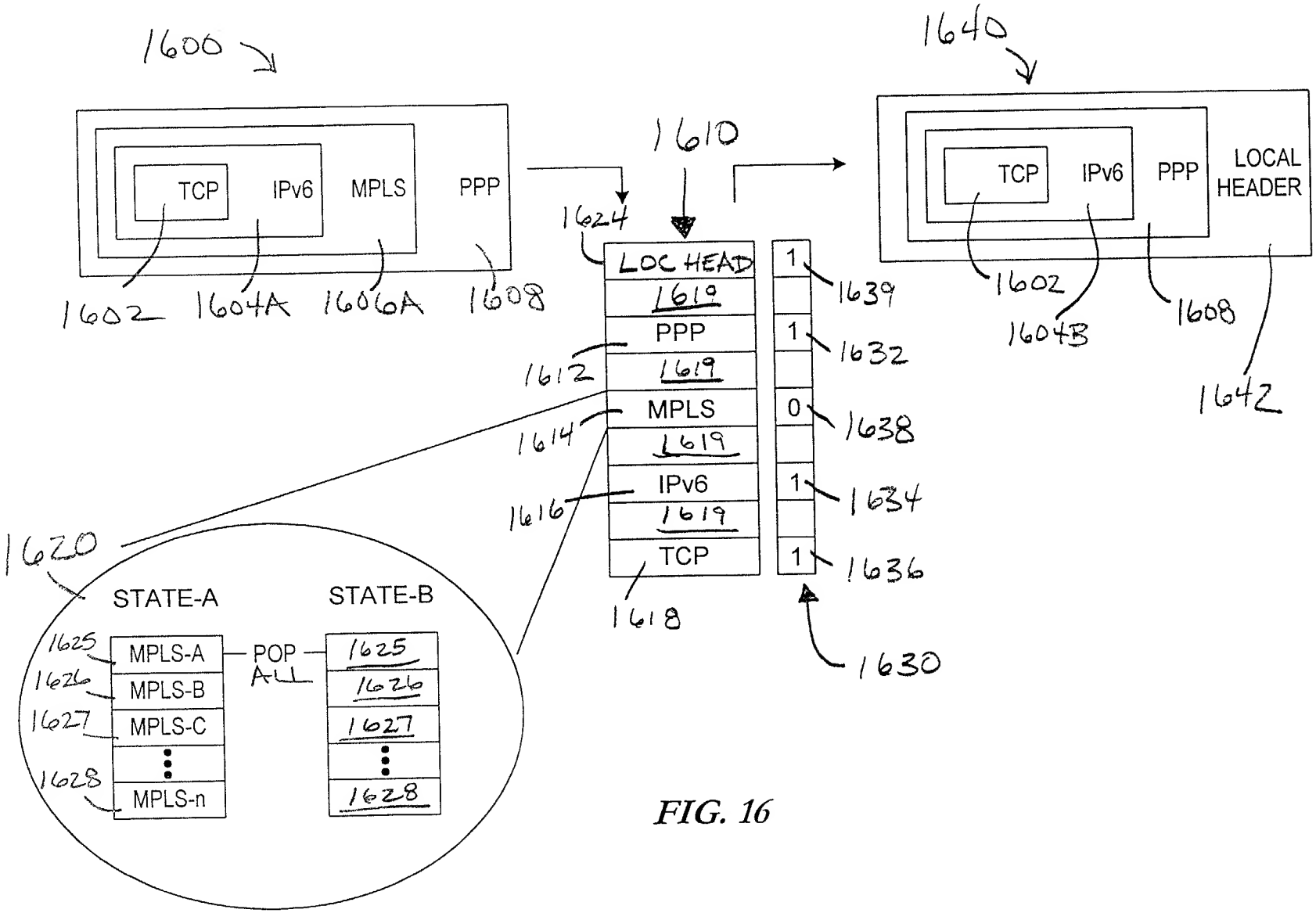

1600
TCP
IPv6
MPLS
PPP
1602
1604A
1606A
1608
1610
1624
LOC HEAD
1619
PPP
1612
MPLS
1614
IPv6
1616
TCP
1618
1
1639
1632
0
1638
1634
1636
1630
1640
LOCAL HEADER
1604B
1642
1620
STATE-A
STATE-B
1625
MPLS-A
POP ALL
1626
MPLS-B
1627
MPLS-C
1628
MPLS-n

FIG. 16

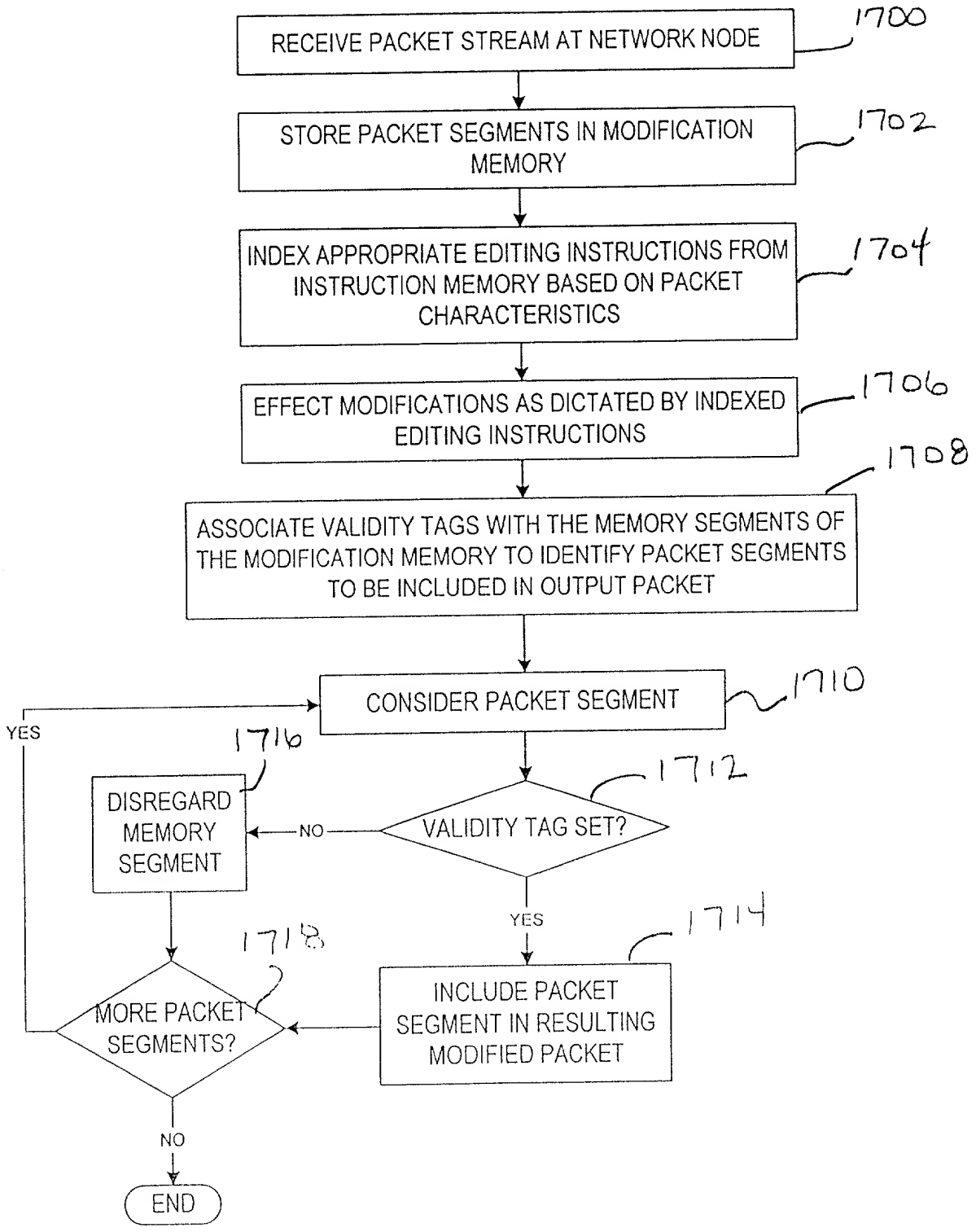

RECEIVE PACKET STREAM AT NETWORK NODE
1700
STORE PACKET SEGMENTS IN MODIFICATION MEMORY
1702
INDEX APPROPRIATE EDITING INSTRUCTIONS FROM INSTRUCTION MEMORY BASED ON PACKET CHARACTERISTICS
1704
EFFECT MODIFICATIONS AS DICTATED BY INDEXED EDITING INSTRUCTIONS
1706
ASSOCIATE VALIDITY TAGS WITH THE MEMORY SEGMENTS OF THE MODIFICATION MEMORY TO IDENTIFY PACKET SEGMENTS TO BE INCLUDED IN OUTPUT PACKET
1708
CONSIDER PACKET SEGMENT
1710
VALIDITY TAG SET?
1712
NO
DISREGARD MEMORY SEGMENT
1716
YES
INCLUDE PACKET SEGMENT IN RESULTING MODIFIED PACKET
1714
MORE PACKET SEGMENTS?
1718
YES
NO
END

FIG. 17